An Efficient Data Structure for Document Clustering Using K-Means Algorithm

Ramanji Killani[1], Suresh Chandra Satapathy[2], and A.M. Sowjanya[3]

[1] MVGR College of Engineering, Vijayanagaram, India
[2] ANITS, Vishakapatnam, India
[3] A.U. College of Engineering, Andhra University, Visakhapatnam, India

Abstract. In this paper, we proposed an efficient data structure called "Sparse Matrices" for representing documents. The document database can be represented by using sparse matrices rather than dense matrices. The matrix can be given as an input for k-means algorithm. Using sparse matrices not only will reduce the size of the database as well as it found efficient in running the program. The experimental results have shown that sparse matrices gives good results compared to dense matrices.

Keywords: Document clustering, sparse matrices and k-means algorithms.

1 Introduction

Text mining is playing crucial role in mining textual databases. Text Mining is a multidisciplinary field, involving information retrieval, text analysis, information extraction, clustering, categorization, visualization, database technology, machine learning and data mining [1],[2]. Text mining data is useful to indicate similarities at the textual level, but is also affected by the ambiguities of vocabularies [9]. The applications of text mining are document clustering, document classification etc. Document clustering techniques have been receiving more and more attentions as a fundamental and enabling tool for efficient organization, navigation, retrieval and summarization of huge volumes of text documents with a good document clustering method; computers can automatically organize a document corpus into a meaningful cluster hierarchy, which enables an efficient browsing and navigation of the corpus [3]. Document clustering is basically a data clustering procedure specialized to the text document materials. The intention of the text document clustering is to divide the given documents into a certain number of groups while maximizing the similarities between the documents into internal groups and minimizing the similarities between the documents disseminated among the different groups [4]. The process of text document clustering can be separated into three parts: document representation, smoothing data in document and clustering. The purpose of document representation is to convert the document data into numerical format and this can be done by using vector space model. Smoothing is used to reduce the size of the data in the document and can be done by using dimensionality reduction techniques. Clustering can be done by using two techniques i.e, partitional clustering and hierarchical clustering, the search space can be partitioned into clusters where as in hierarchical, the space can be

S.C. Satapathy et al. (Eds.): Proceedings of the InConINDIA 2012, AISC 132, pp. 337–343.
springerlink.com

divided into hierarchical forms. In this work, we have used k-means algorithm for clustering.

Generally, the dense matrix occupies lots of space and k-means algorithm is taking lots of time to give the result. The sparse matrices will take less space and hence the dataset will be in compressed form. So, the mining process with k-means algorithm will run faster.

In this work, we are using document database of huge size and the data is not dense as well. Due to that reason, the database is represented using sparse matrices. The advantage of using sparse matrices is by nature the data is easily compressed, so the space can be reduced and as well as the algorithm runs faster.

The rest of the paper is organized as follows: Section-2 gives basic idea of document representation. In section-3, the sparse matrices are discussed. Clustering algorithm is presented in section-4. The simulation results are given in section-5. Finally the paper is concluded in section-6.

2 Document Representation

2.1 Data Representation for Document Clustering

To find the relevance in the documents, the dataset is represented using vector space model. Each document is represented as a vector and the document database is treated as a vector space. The documents can be represented into vectors in two phases. In the first phase the documents are scanned. This phase identifies the unique words in the document dataset and gives unique term number for each term in the document. In the second phase, the documents are represented in vectors, i.e. with their dimension and the quantity of the dimension. A document *d* is represented as follows, *d*= {*t1, t2, t3…tn*} where *t1, t2…tn* are terms.

$$d= a_1t_1+a_2t_2+\ldots+a_nt_n \quad (1)$$

Where a_1, a_2,…., a_n are frequencies of terms i.e, how many times terms t_1,t_2,…,t_n are repeated in the document.

2.2 Picking Important Terms in Documents

Important terms are to be identified before applying clustering techniques. To identify important terms a measure called term-frequency and inverse document frequency (Tf-Idf) is used. The weight of term '*x*' in a document is given by the following equation (2):

$$w_{yx} = tf_{yx} * idf_{yx} \quad (2)$$

2.3 Finding Similar Documents

There are number of methods to calculate similarity between two documents such as Cosine methods, Euclidian method, Minkowski distance measure etc. In our work we have used Minkowski distance measure as it is the generalization of both Euclidian and Manhattan distance.

$$d(m_i, m_j) = \sqrt{\sum_{i=1}^{dm} (m_{ik} - m_{jk})^2 \Big/ d_m} \quad (3)$$

Where m_i, m_j are any two document vectors, dm is the dimension of vector space and m_{ik} , m_{jk} are weights for dimension 'k'.

3 Sparse Matrices

The tabular data can be represented in matrices format in computer's memory. There are two kinds of matrices, sparse matrices and dense matrices. The sparse matrices consists of many elements as zeros where as in dense matrices consists of many elements as non zeros. The examples sparse matrices are Identity matrix. A matrix having only a small percentage of nonzero elements is said to be sparse. In a practical sense an n X n matrix is classified as sparse, if it has order of n nonzero elements; say two to ten nonzero elements in each row, for large n. The matrices associated with a large class of man-made systems are sparse. For example, the matrix representing the communication paths of the employees in a large organization is sparse, provided that the ith row and the jth column element of the matrix is nonzero if and only if employees i and j interact.

Sparse matrices appear in Information retrieval, linear programming, structural analyses, network theory and power distribution systems, numerical solution of differential equations, graph theory, genetic theory, social and behavioral sciences and computer programming. It is difficult to solve the problems which involves with large matrices which either are impossible to invert on available computer storage or are very expensive to invert. Since such matrices are generally sparse it is useful to know the techniques currently available for dealing with sparse matrices. This allows one to choose the best technique for the type of sparse matrix encounters. The time and effort required to develop the various techniques for handling sparse matrices is especially justified when several matrices having the same zero-nonzero structures but differing numerical values have to be handled [8]. Dense matrices are unsuitable for representing high dimensional data, there is a need for a collection of matrices from real applications where data can be represented by using sparse matrices [7]. Generally, the data in sparse matrices can be represented as row, column and value of the element.

Sparse matrix formats compress large matrices with a small number of nonzero elements into a more compact representation. The goal is to both reduce memory footprint and increase efficiency of operations such as sparse matrix-vector multiplication (SpMV) [10].

For example, let's take I4 matrix. The dense matrix representation for I4 is as follows:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The sparse matrix representation for I4 is given below:

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \\ 3 & 3 & 1 \end{bmatrix}$$

3.1 Memory Comparison between Sparse Matrices and Dense Matrices

The dense matrix occupies huge amount memory especially, in the application of document mining. Let us consider a 500 x 500 matrix with 1994 nonzero elements. The dense matrix will require 500 x 500 x 4 = 1million bytes where as the sparse matrix will takes 3 x 1994 x 4= 23,928 bytes.

3.2 Representation of Sparse Matrix

Here the entire document is represented in sparse matrix vector where each document is a sparse matrix. Let us assume, the document database consists of (d1, d2, …, dn) documents. Each document is represented as follows:

$$d1 = \begin{bmatrix} col1 & col2.. & col\ln \\ & & \\ val1 & val2.. & val\ln \end{bmatrix} \tag{4}$$

Where col1, col2, … and coln are column values and row value is 1 and the element values are val1, val2, …, valn. Here, only non zero values will be stored.

4 Clustering Algorithm

Clustering algorithms are classified into partitional clustering and hierarchical clustering. In this work, we have used K-means Algorithm for document clustering.

4.1 K-Means Algorithm

K-Means algorithm [5] is a partitional-based clustering algorithm. It searches for the solution in local space area. K-Means algorithm divides the problem space area into partitions and searches for the solution. The algorithm takes the input in sparse matrix format. The similarity matrix consists of Tf-Idf values, number of clusters '*n*' and initial centroids. The output is in terms of Set of clusters, Intra-cluster distance, Inter-cluster distance. K-means has some disadvantages like the solution is completely depends upon the initial cluster centroids which are generated in random manner and it searches for the solution in local space area and gets trapped in local optima often.

The K-means algorithm can be summarized as follows:

(1) Randomly select cluster centroid vectors to set an initial dataset partition.
(2) Assign each document vector to the closest cluster centroids.

(3) Recalculate the cluster centroid vector **cj** using equation 5.

$$C_j = \frac{1}{n_j} \sum_{\forall d_j \in S_j} d_j \tag{5}$$

where **dj** denotes the document vectors that belong to cluster **Sj**; **cj** stands for the centroid vector; **nj** is the number of document vectors that belong to cluster **Sj**.

(4) Repeat step 2 and 3 until the convergence is achieved.

5 Experimental Setup and Results

This experiment is performed by using J2SE 6.0. The clusters are developed with K-means algorithm. The three datasets are taken from Tech TC-300 repository [6]. The descriptions are given below:

1. Exp_240218_474717 consists of 185 documents and 6560 terms: data1
2. Exp_22294_25575 consists of 127 documents and 12812 terms: data2
3. Exp_20673_269078 consists of 147 documents and 14600 terms: data3

The numbers of clusters are chosen by user based on the knowledge derived from the contents of the dataset under investigation. For all experiments we have chosen number of clusters ranging from 3 to 7. The good cluster should have minimum intra cluster and maximum inter cluster distances.

The results shown below are of document clustering implemented with dense matrix.

Table 1. Results of Documenter clustering using Dense Matrix

S. No	Dataset Size		Number of Clusters	Space required	Average Intra Cluster Distance	Average Inter Cluster Distance	Time (in milli seconds)
	Number of Documents	Size of dimension					
1	185	6560	3	1.16MB	9.374	13.584	6831.2
2	185	6560	4	1.16MB	8.868	18.341	6945.5
3	185	6560	5	1.16MB	7.309	14.164	6739.4
4	185	6560	6	1.16MB	9.635	18.513	7254.6
5	185	6560	7	1.16MB	6.856	15.640	7139.2
6	127	12812	3	6.21MB	7.441	14.456	8247.5
7	127	12812	4	6.21MB	8.865	16.774	9000.0
8	127	12812	5	6.21MB	6.911	14.449	8894.7
9	127	12812	6	6.21MB	7.207	16.446	9575.0
10	127	12812	7	6.21MB	6.392	14.637	9569.3
11	147	14600	3	8.19MB	8.787	21.162	10650.5
12	147	14600	4	8.19MB	8.158	15.780	10529.7
13	147	14600	5	8.19MB	5.062	13.452	10887.3
14	147	14600	6	8.19MB	5.787	15.848	11686.2
15	147	14600	7	8.19MB	5.883	16.305	11396.8

The results shown below are of document clustering implemented with sparse matrix.

Table 2. Results of Documenter clustering using Sparse Matrix

S. No	Dataset Size		Number of Clusters	Space required	Space occupied	Average Intra Cluster Distance	Average Inter Cluster Distance	Time (in milli second)
	Number of Documents	Size of dimension						
1	185	6560	3	1.16MB	0.35MB	8.32164	12.34232	7678.8
2	185	6560	4	1.16MB	0.35MB	9.65678	16.03120	7137.8
3	185	6560	5	1.16MB	0.35MB	7.30900	14.16478	7087.4
4	185	6560	6	1.16MB	0.35MB	7.67342	19.23376	7378.2
5	185	6560	7	1.16MB	0.35MB	8.4267	15.64021	7099.8
6	127	12812	3	6.21MB	0.24MB	8.12150	12.45698	8578.0
7	127	12812	4	6.21MB	0.24MB	7.86576	16.77452	9034.4
8	127	12812	5	6.21MB	0.24MB	5.39230	14.13302	9109.8
9	127	12812	6	6.21MB	0.24MB	6.86863	13.09628	9393.6
10	127	12812	7	6.21MB	0.24MB	7.42410	12.45108	9359.2
11	147	14600	3	8.19MB	0.35MB	9.21124	16.57858	10880.8
12	147	14600	4	8.19MB	0.35MB	7.34905	15.64275	10768.8
13	147	14600	5	8.19MB	0.35MB	4.83864	13.20175	11100.0
14	147	14600	6	8.19MB	0.35MB	5.28120	14.67351	11453.2
15	147	14600	7	8.19MB	0.35MB	5.21360	12.32568	11694.0

6 Conclusion

By using the sparse matrix, the documents data can be represented. Sparse matrix representation occupies only less memory to store the entire document description. The results are showing that the dense matrix occupies lot of memory compared to sparse matrix. Therefore, the sparse matrix is the good data structure to represent the high dimensional data.

References

1. Tan, A.-H.: Text Mining state of art and challenges. In: Proceedings of the PAKDD 1999 Workshop (1999)
2. Han, J., Kamber, M.: DataMining concepts and Techniques, 2nd edn. Morgan Kaufmann publishers (2006)
3. van der Maaten, L.J.P., Postma, E.O., van den Herik, H.J.: Dimensionality Reduction a Comparative Review. Citeseer (2007)
4. Cui, X., Potok, T.E., Palathingal, P.: Document Clustering using particle swarm optimization. In: IEEE Swarm Intelligence Symposium (2005)
5. MacQueen, J.B.: Some Methods for classification and Analysis of Multivariate Observations. In: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press (1967)

6. The TechTC-300 Test Collection for Text Categorization Version: 1.0 TechTC - Technion Repository of Text Categorization Datasets, Maintained by: Evgeniy Gabrilovich gabr@cs.technion.ac.il
7. Davis, T.A.: The University of Florida Sparse Matrix Collection
8. Tewarson, R.P.: Sparse matrices. ELSEVIER
9. Liu, X., Yu, S., Moreau, Y., De Moor, B., Glänzel, W., Janssens, F.: Hybrid Clustering of Text Mining and Bibliometrics Applied to Journal Sets. In: Siam Proceeding on Data Mining (2009)
10. Arnold, G., Holzl, J., Koksal, A.S., Bodík, R., Sagiv, M.: Specifying and Verifying Sparse Matrix Codes. In: ICFP 2010 Proceedings of the 15th ACM SIGPLAN International Conference on Functional Programming (2010)